

Documentacion sobre:

Sistema de Gestión de Reservas para un Hotel



Realizado por: Rene Cruz Montejo

Herramienta utilizada:

MySQL workbench

Introducción:

Sistema de Gestión de Reservas para un Hotel

Requerimiento: Desarrollar una base de datos que permita gestionar las reservas, clientes, habitaciones y servicios adicionales en un hotel. El sistema debe facilitar la asignación de habitaciones, el registro de entradas y salidas, y la gestión de servicios como restaurante, spa y conferencias.

Aspectos de Entrega:

- **Propuesta Justificación:** Describir los beneficios del sistema de gestión de reservas para mejorar la eficiencia operativa y la experiencia del cliente en el hotel.

- **Diagrama Entidad-Relación (DER) en 3FN:** Representar visualmente la estructura de la base de datos, asegurando la eliminación de redundancias y optimización del diseño.

- **Script SQL:** Incluir la creación de todas las tablas, relaciones, claves

- **Implementación de Índices, Vistas y Procedimientos Almacenados:**

Mejorar el rendimiento y la funcionalidad del sistema con índices para búsquedas rápidas, vistas para consultas frecuentes y procedimientos almacenados para operaciones como check-in/check-out y reserva de servicios.

- **Procedimientos Almacenados Específicos:**

1. **Check-in de Cliente:** Registra la llegada de un cliente, asigna habitación y servicios solicitados.

2. **Check-out de Cliente:** Gestiona la salida del cliente, procesa pagos y actualiza la disponibilidad de la habitación.

3. Reserva de Servicios: Permite la reserva de servicios adicionales por parte de los clientes.

- **Manejo de Errores en Procedimientos Almacenados:** Implementar

Manejo de errores para asegurar la fiabilidad y la integridad de los datos.

- **Generación de registros “prueba”, al menos 1000 registros en cada una de las**

tablas.

- **Requerimiento de Trigger en el Sistema de Reservas para un Hotel**

Con el objetivo de automatizar ciertas acciones y mantener la integridad de los datos

Requerimiento de Seguridad en el Sistema de Reservas para un Hotel:

- Implementar medidas de seguridad para proteger la información sensible y confidencial de los clientes y usuarios del sistema, aplicando las siguientes técnicas:

- **Cifrado de datos confidenciales:** Cifrar la información sensible almacenada en la base de datos, como detalles de clientes, para evitar accesos no autorizados.

- **Hash de contraseñas:** Aplicar algoritmos de hash como SHA-256 para almacenar de forma segura las contraseñas de los usuarios, evitando que se almacenen en texto plano.

- **Trigger de cifrado:** Configurar triggers en la base de datos que garanticen que ciertos datos siempre se almacenen cifrados, asegurando su protección ante posibles brechas de seguridad.

- **Cifrado de respaldos de la base de datos:** Proteger los archivos de respaldo mediante cifrado, para evitar el acceso no autorizado en caso de que los

respaldos sean manipulados fuera del sistema.

Desarrollo:

Propósito

El script crea una base de datos llamada HotelManagement diseñada para gestionar la información de un sistema hotelero, incluyendo clientes, habitaciones, reservas, servicios, pagos y relaciones entre estos elementos.

Beneficios del sistema:

1. Mejora de la eficiencia operativa mediante la automatización de tareas como asignación de habitaciones y reservas de servicios.
2. Reducción de errores manuales, garantizando una gestión precisa y confiable.
3. Incremento de la satisfacción del cliente mediante una experiencia personalizada y fluida.

Script SQL para la creación de tablas y relaciones

```
1      -- Script SQL para la creación de tablas y relaciones
2 •    CREATE DATABASE HotelManagement;
3 •    USE HotelManagement;
4
```

- CREATE DATABASE HotelManagement: Crea una nueva base de datos llamada HotelManagement.
- USE HotelManagement: Selecciona esta base de datos para trabajar con ella en las operaciones posteriores.

Creación de tabla cliente

```

5      -- Tabla Cliente
6  ● ○ CREATE TABLE Cliente (
7      ClienteID INT AUTO_INCREMENT PRIMARY KEY,
8      Nombre VARCHAR(100) NOT NULL,
9      Apellido VARCHAR(100) NOT NULL,
10     Email VARCHAR(150) UNIQUE NOT NULL,
11     Telefono VARCHAR(15),
12     FechaRegistro DATETIME DEFAULT CURRENT_TIMESTAMP
13 );

```

- ClienteID: Identificador único para cada cliente. Es clave primaria y se genera automáticamente.
- Nombre y Apellido: Almacenan los nombres del cliente. Son campos obligatorios (NOT NULL).
- Email: Dirección de correo del cliente. Es único (UNIQUE) y obligatorio.
- Telefono: Número de teléfono del cliente. Es opcional.
- FechaRegistro: Fecha y hora en que el cliente fue registrado. Se llena automáticamente con la fecha actual.

Creación de tabla habitación

```

15     -- Tabla Habitación
16  ● ○ CREATE TABLE Habitacion (
17     HabitacionID INT AUTO_INCREMENT PRIMARY KEY,
18     NumeroHabitacion VARCHAR(10) UNIQUE NOT NULL,
19     Tipo VARCHAR(50) NOT NULL,
20     PrecioPorNoche DECIMAL(10, 2) NOT NULL,
21     Estado ENUM('Disponible', 'Ocupada', 'Mantenimiento') DEFAULT 'Disponible'
22 );

```

- HabitacionID: Identificador único para cada habitación.
- NumeroHabitacion: Número único de habitación, obligatorio y no repetible.
- Tipo: Tipo de habitación (ejemplo: sencilla, doble, suite).

- PrecioPorNoche: Precio por noche. Utiliza el tipo DECIMAL para valores con dos decimales.
- Estado: Indica si la habitación está Disponible, Ocupada o en Mantenimiento.

Creación de tabla servicio

```

24      -- Tabla Servicio
25  ● ○ CREATE TABLE Servicio (
26      ServicioID INT AUTO_INCREMENT PRIMARY KEY,
27      Nombre VARCHAR(100) NOT NULL,
28      Precio DECIMAL(10, 2) NOT NULL
29  );

```

- ServicioID: Identificador único para cada servicio.
- Nombre: Nombre del servicio (ejemplo: desayuno, spa).
- Precio: Costo del servicio.

Creación de tabla reserva

```

31      -- Tabla Reserva
32  ● ○ CREATE TABLE Reserva (
33      ReservaID INT AUTO_INCREMENT PRIMARY KEY,
34      ClienteID INT NOT NULL,
35      HabitacionID INT NOT NULL,
36      FechaInicio DATE NOT NULL,
37      FechaFin DATE NOT NULL,
38      Total DECIMAL(10, 2),
39      FOREIGN KEY (ClienteID) REFERENCES Cliente(ClienteID),
40      FOREIGN KEY (HabitacionID) REFERENCES Habitacion(HabitacionID)
41  );

```

- ReservaID: Identificador único para cada reserva.
- ClienteID: Enlace al cliente que realiza la reserva (clave foránea).

- HabitaciónID: Habitación reservada (clave foránea).
- FechaInicio y FechaFin: Fechas de inicio y fin de la reserva.
- Total: Total calculado para la reserva.
- FOREIGN KEY: Define las relaciones con las tablas Cliente y Habitación.

Creación de tabla pago

```

43      -- Tabla Pago
44  ● ○ CREATE TABLE Pago (
45      PagoID INT AUTO_INCREMENT PRIMARY KEY,
46      ReservaID INT NOT NULL,
47      Monto DECIMAL(10, 2) NOT NULL,
48      MetodoPago ENUM('Tarjeta', 'Efectivo', 'Transferencia') NOT NULL,
49      FechaPago DATETIME DEFAULT CURRENT_TIMESTAMP,
50      FOREIGN KEY (ReservaID) REFERENCES Reserva(ReservaID)
51  );

```

- PagoID: Identificador único para cada pago.
- ReservaID: Referencia a la reserva pagada (clave foránea).
- Monto: Monto del pago.
- MetodoPago: Método usado para el pago (Tarjeta, Efectivo o Transferencia).
- FechaPago: Fecha y hora del pago.

Creación de tabla servicios_reservas (relación muchos a muchos)

```

53      -- Tabla Servicios_Reservas (relación muchos a muchos)
54  ● ○ CREATE TABLE Servicios_Reservas (
55      ID INT AUTO_INCREMENT PRIMARY KEY,
56      ReservaID INT NOT NULL,
57      ServicioID INT NOT NULL,
58      FOREIGN KEY (ReservaID) REFERENCES Reserva(ReservaID),
59      FOREIGN KEY (ServicioID) REFERENCES Servicio(ServicioID)
60  );

```

- ID: Identificador único para cada relación.

- ReservaID y ServicioID: Relación muchos a muchos entre reservas y servicios.

Índices

```

62      -- Índices
63 •    CREATE INDEX idx_cliente_email ON Cliente(Email);
64 •    CREATE INDEX idx_habitacion_estado ON Habitacion(Estado);
65 •    CREATE INDEX idx_reserva_fechas ON Reserva(FechaInicio, FechaFin);

```

- Se crean índices para acelerar búsquedas en campos clave como Email, Estado y FechaInicio/FechaFin.

Procedimientos almacenados

```

67      -- Procedimientos Almacenados
68      DELIMITER //
69 •    CREATE PROCEDURE CheckInCliente(IN p_ClienteID INT, IN p_HabitacionID INT, IN p_FechaInicio DATE, IN p_FechaFin DATE, OUT p_Resultado VARCHAR(100))
70      BEGIN
71          DECLARE v_Disponible ENUM('Disponible', 'Ocupada', 'Mantenimiento');
72
73          SELECT Estado INTO v_Disponible
74          FROM Habitacion
75          WHERE HabitacionID = p_HabitacionID;
76
77          IF v_Disponible = 'Disponible' THEN
78              INSERT INTO Reserva (ClienteID, HabitacionID, FechaInicio, FechaFin, Total)
79              VALUES (p_ClienteID, p_HabitacionID, p_FechaInicio, p_FechaFin, 0);
80
81              UPDATE Habitacion
82              SET Estado = 'Ocupada'
83              WHERE HabitacionID = p_HabitacionID;
84
85              SET p_Resultado = 'Check-in exitoso';
86          ELSE
87              SET p_Resultado = 'Habitación no disponible';
88          END IF;
89      END //
90      DELIMITER ;
91
92      DELIMITER //
93 •    CREATE PROCEDURE CheckOutCliente(IN p_ReservaID INT, OUT p_Resultado VARCHAR(100))
94      BEGIN
95          DECLARE v_HabitacionID INT;
96
97          SELECT HabitacionID INTO v_HabitacionID
98          FROM Reserva

```



```

99         WHERE ReservaID = p_ReservaID;
100
101     UPDATE Habitacion
102     SET Estado = 'Disponible'
103     WHERE HabitacionID = v_HabitacionID;
104
105     DELETE FROM Reserva
106     WHERE ReservaID = p_ReservaID;
107
108     SET p_Resultado = 'Check-out exitoso';
109 END //
110 DELIMITER ;

```

- Check-in del cliente:

Valida si la habitación está disponible.

Registra una nueva reserva y actualiza el estado de la habitación.

- Check-out del cliente:

Libera la habitación y elimina la reserva.

Seguridad

```

112 • -- Seguridad
113 -- Cifrado de datos sensibles
114 ALTER TABLE Cliente ADD COLUMN EmailCifrado VARBINARY(255);
115 • UPDATE Cliente SET EmailCifrado = AES_ENCRYPT(Email, 'clave_segura');
116 -- Hash de contraseñas
117 • ALTER TABLE Cliente ADD COLUMN PasswordHash VARBINARY(255);
118
119 -- Trigger para cifrado automático
120 DELIMITER //
121 • CREATE TRIGGER TriggerCifrado BEFORE INSERT ON Cliente
122   FOR EACH ROW
123   BEGIN
124       SET NEW.EmailCifrado = AES_ENCRYPT(NEW.Email, 'clave_segura');
125   END //
126 DELIMITER ;

```

- Cifrado de Email:

Añade una columna para almacenar emails cifrados usando

AES_ENCRYPT.

- Trigger para cifrado automático:

Cifra automáticamente los emails al insertar un nuevo cliente.

Script para la generación de registros

```
1  -- Script para la generacion de registros
2  •  DROP PROCEDURE IF EXISTS GenerarDatosPrueba;
```

- DROP PROCEDURE IF EXISTS: Elimina el procedimiento almacenado llamado GenerarDatosPrueba si ya existe, para evitar conflictos al recrearlo.

```
4  -- Desactivar temporalmente las restricciones de claves foráneas
5  •  SET foreign_key_checks = 0;
```

- SET foreign_key_checks = 0: Desactiva temporalmente las verificaciones de claves foráneas, lo que permite eliminar o insertar datos en cualquier orden sin violar las restricciones.

```
7  -- Eliminar los registros de las tablas dependientes en el orden correcto
8  •  TRUNCATE TABLE Servicios_Reservas;
9  •  TRUNCATE TABLE Pago;
10 •  TRUNCATE TABLE Reserva;
11 •  TRUNCATE TABLE Habitacion;
12 •  TRUNCATE TABLE Cliente;
13 •  TRUNCATE TABLE Servicio;
```

- TRUNCATE TABLE: Elimina todos los registros de cada tabla de manera rápida, reseteando los contadores de autoincremento. Se realiza en el orden correcto para evitar problemas con las claves foráneas.

```
15      -- Reactivar las restricciones de claves foráneas
16 •    SET foreign_key_checks = 1;
```

- SET foreign_key_checks = 1: Reactiva las verificaciones de claves foráneas después de la limpieza.

Procedimiento almacenado para generar datos de prueba

```
19      DELIMITER //
```

```
20      |
```

```
21 •    CREATE PROCEDURE GenerarDatosPrueba()
```

```
22      BEGIN
```

```
23          DECLARE i INT DEFAULT 1;
```

```
24          DECLARE v_NumeroHabitacion VARCHAR(10);
```

```
25          DECLARE v_ClienteID INT;
```

```
26          DECLARE v_HabitacionID INT;
```

```
27          DECLARE v_ReservaID INT;
```

```
28          DECLARE v_ServicioID INT;
```

```
29          DECLARE v_Email VARCHAR(100);
```

- CREATE PROCEDURE GenerarDatosPrueba(): Crea un procedimiento almacenado llamado GenerarDatosPrueba.
- DECLARE: Declara variables locales para usar dentro del procedimiento.

Estas variables son:

i: Contador para los bucles.

v_NumeroHabitacion, v_Email: Para generar datos específicos como números de habitación o correos electrónicos.

v_ClienteID, v_HabitacionID, v_ReservaID, v_ServicioID: Para guardar temporalmente los identificadores generados aleatoriamente.

Inserción de datos en la tabla Cliente

```

31      -- Insertar datos de clientes
32      WHILE i <= 1000 DO
33          SET v_Email = CONCAT('cliente', LPAD(i, 4, '0'), '_', FLOOR(RAND() * 10000), '@dominio.com');
34
35          -- Insertar cliente
36          INSERT INTO Cliente (Nombre, Apellido, Email, Telefono)
37          VALUES (CONCAT('Nombre', i), CONCAT('Apellido', i), v_Email, CONCAT('123456', LPAD(i, 4, '0')));
38
39          SET i = i + 1;
40      END WHILE;

```

- Crea 1000 clientes con:
- Email único: Generado combinando un contador, un número aleatorio, y un dominio.
- Teléfono único: Generado usando un número secuencial.
- Utiliza un bucle WHILE para realizar las inserciones repetitivas.

Inserción de datos en la tabla Habitación

```

42      -- Insertar datos de habitaciones con números secuenciales garantizados
43      SET i = 1;
44      WHILE i <= 1000 DO
45          SET v_NumeroHabitacion = CONCAT('HAB', LPAD(i, 4, '0')); -- Números secuenciales de habitación como HAB0001, HAB0002, etc.
46
47          -- Insertar habitación
48          INSERT INTO Habitacion (NumeroHabitacion, Tipo, PrecioPorNoche)
49          VALUES (v_NumeroHabitacion, 'Individual', ROUND(RAND() * 100 + 50, 2));
50
51          SET i = i + 1;
52      END WHILE;

```

- Inserta 1000 habitaciones:
- Número secuencial único: Como HAB0001, HAB0002, etc.
- Tipo: Fijado como Individual.
- Precio por noche: Aleatorio entre 50 y 150.

Insertión de datos en la tabla Servicio

```

54      -- Insertar datos de servicios
55      SET i = 1;
56      WHILE i <= 1000 DO
57          INSERT INTO Servicio (Nombre, Precio)
58          VALUES (CONCAT('Servicio', i), ROUND(RAND() * 100 + 20, 2));
59          SET i = i + 1;
60      END WHILE;

```

- Inserta 1000 servicios:
- Nombre único: Servicio1, Servicio2, etc.
- Precio aleatorio: Entre 20 y 120.

Insertión de datos en la tabla Reserva

```

62      -- Insertar datos de reservas
63      SET i = 1;
64      WHILE i <= 1000 DO
65          SET v_ClienteID = FLOOR(RAND() * 1000) + 1;
66          SET v_HabitacionID = FLOOR(RAND() * 1000) + 1;
67
68          -- Insertar reserva
69          INSERT INTO Reserva (ClienteID, HabitacionID, FechaInicio, FechaFin, Total)
70          VALUES (v_ClienteID, v_HabitacionID, CURDATE(), DATE_ADD(CURDATE(), INTERVAL (FLOOR(RAND() * 7) + 1) DAY), ROUND(RAND() * 500 + 100, 2));
71
72          SET i = i + 1;
73      END WHILE;

```

- Inserta 1000 reservas con:
- Cliente y habitación aleatorios.
- Fechas: La actual como inicio y hasta 7 días más.
- Total aleatorio: Entre 100 y 600.

Insertión de datos en la tabla Pago

```

75  -- Insertar datos de pagos
76  SET i = 1;
77  WHILE i <= 1000 DO
78      SET v_ReservaID = FLOOR(RAND() * 1000) + 1;
79
80      -- Insertar pago
81      INSERT INTO Pago (ReservaID, Monto, MetodoPago)
82      VALUES (v_ReservaID, ROUND(RAND() * 500 + 100, 2), CASE WHEN FLOOR(RAND() * 3) = 0 THEN 'Tarjeta' WHEN FLOOR(RAND() * 3) = 1 THEN 'Efectivo' ELSE 'Transferencia' END);
83
84      SET i = i + 1;
85  END WHILE;

```

- Inserta 1000 pagos con:
- Reserva aleatoria.
- Monto aleatorio: Entre 100 y 600.
- Método de pago aleatorio.

Insertión de datos en la tabla Servicios_Reservas

```

87      -- Insertar datos de servicios en reservas
88      SET i = 1;
89      WHILE i <= 1000 DO
90          SET v_ReservaID = FLOOR(RAND() * 1000) + 1;
91          SET v_ServicioID = FLOOR(RAND() * 1000) + 1;
92
93          -- Insertar servicios en reservas
94          INSERT INTO Servicios_Reservas (ReservaID, ServicioID)
95          VALUES (v_ReservaID, v_ServicioID);
96
97          SET i = i + 1;
98      END WHILE;
99
100  END //
101

```

- Inserta relaciones entre reservas y servicios:
- Reserva y servicio aleatorios.

Llamada al procedimiento

```

104 •  -- Llamar al procedimiento para generar datos de prueba
105     CALL GenerarDatosPrueba();














```

Ejecuta el procedimiento para generar todos los datos en las tablas.

En las siguientes imágenes podemos observar la creación de los registros (1000).

Observamos que el método utilizado para cifrar el email en el script es AES_ENCRYPT, una función integrada en MySQL que emplea el algoritmo de cifrado AES (Advanced Encryption Standard)

controlBD1 GeneradorRegistros* registroprueba mostrartablas mostrarClientes registrosClientes x reg

        Don't Limit     

1 • `SELECT * FROM Cliente LIMIT 1000;`

2

3

Result Grid							
		Filter Rows:		Edit:		Export/Import:	
ClienteID	Nombre	Apellido	Email	Telefono	FechaRegistro	EmailCifrado	
51	Nombre51	Apellido51	cliente0051_3319@dominio.com	1234560051	2025-01-16 02:03:20	NULL	
52	Nombre52	Apellido52	cliente0052_9983@dominio.com	1234560052	2025-01-16 02:03:20	NULL	
53	Nombre53	Apellido53	cliente0053_9955@dominio.com	1234560053	2025-01-16 02:03:20	NULL	
54	Nombre54	Apellido54	cliente0054_9829@dominio.com	1234560054	2025-01-16 02:03:20	NULL	
55	Nombre55	Apellido55	cliente0055_9281@dominio.com	1234560055	2025-01-16 02:03:20	NULL	
56	Nombre56	Apellido56	cliente0056_6919@dominio.com	1234560056	2025-01-16 02:03:20	NULL	
57	Nombre57	Apellido57	cliente0057_6750@dominio.com	1234560057	2025-01-16 02:03:20	NULL	
58	Nombre58	Apellido58	cliente0058_2996@dominio.com	1234560058	2025-01-16 02:03:20	NULL	
59	Nombre59	Apellido59	cliente0059_4727@dominio.com	1234560059	2025-01-16 02:03:20	NULL	
60	Nombre60	Apellido60	cliente0060_4651@dominio.com	1234560060	2025-01-16 02:03:20	NULL	
61	Nombre61	Apellido61	cliente0061_9075@dominio.com	1234560061	2025-01-16 02:03:20	NULL	
62	Nombre62	Apellido62	cliente0062_1419@dominio.com	1234560062	2025-01-16 02:03:20	NULL	
63	Nombre63	Apellido63	cliente0063_9873@dominio.com	1234560063	2025-01-16 02:03:20	NULL	
64	Nombre64	Apellido64	cliente0064_5109@dominio.com	1234560064	2025-01-16 02:03:20	NULL	
65	Nombre65	Apellido65	cliente0065_5926@dominio.com	1234560065	2025-01-16 02:03:20	NULL	
66	Nombre66	Apellido66	cliente0066_4302@dominio.com	1234560066	2025-01-16 02:03:20	NULL	
67	Nombre67	Apellido67	cliente0067_3732@dominio.com	1234560067	2025-01-16 02:03:20	NULL	
68	Nombre68	Apellido68	cliente0068_5756@dominio.com	1234560068	2025-01-16 02:03:20	NULL	

controlBD1 GeneradorRegistros* registroprueba mostrartablas mostrarClientes registrosClientes registrosHabitacion x registrosreserva

Don't Limit

1 • SELECT * FROM Habitacion LIMIT 100;

2

<

Result Grid

Filter Rows:










Edit:

Export/Import:






Wrap Cell Content:

	HabitacionID	NumeroHabitacion	Tipo	PrecioPorNoche	Estado
▶	1	HAB0001	Individual	122.27	Disponible
	2	HAB0002	Individual	65.18	Disponible
	3	HAB0003	Individual	109.10	Disponible
	4	HAB0004	Individual	99.93	Disponible
	5	HAB0005	Individual	122.37	Disponible
	6	HAB0006	Individual	62.04	Disponible
	7	HAB0007	Individual	93.11	Disponible
	8	HAB0008	Individual	129.44	Disponible
	9	HAB0009	Individual	117.85	Disponible
	10	HAB0010	Individual	50.95	Disponible
	11	HAB0011	Individual	51.17	Disponible
	12	HAB0012	Individual	53.03	Disponible
	13	HAB0013	Individual	61.62	Disponible
	14	HAB0014	Individual	99.03	Disponible
	15	HAB0015	Individual	60.28	Disponible
	16	HAB0016	Individual	54.31	Disponible
	17	HAB0017	Individual	140.72	Disponible
	18	HAB0018	Individual	90.65	Disponible

controlBD1 GeneradorRegistros* registroprueba mostrartablas mostrarClientes registrosClientes registrosHabitacion registrosreserva x registrosServicio registrosPago revisarCheck



Don't Limit



```
1
2 • SELECT * FROM Reserva LIMIT 100;
3
```

Result Grid

Filter Rows:

Edits










Export/Import:

Wrap Cell Content:






ReservaID	ClienteID	HabitacionID	FechaInicio	FechaFin	Total
1	149	390	2025-01-16	2025-01-20	263.96
2	143	730	2025-01-16	2025-01-18	553.29
3	877	661	2025-01-16	2025-01-21	297.28
4	947	551	2025-01-16	2025-01-23	550.45
5	774	167	2025-01-16	2025-01-20	124.13
6	714	422	2025-01-16	2025-01-23	390.50
7	998	243	2025-01-16	2025-01-18	286.44
8	205	906	2025-01-16	2025-01-23	519.63
9	464	801	2025-01-16	2025-01-21	431.75
10	478	399	2025-01-16	2025-01-20	400.73
11	329	841	2025-01-16	2025-01-18	377.68
12	131	988	2025-01-16	2025-01-20	483.99
13	202	703	2025-01-16	2025-01-23	315.59
14	434	873	2025-01-16	2025-01-17	444.00
15	258	224	2025-01-16	2025-01-19	128.97
16	253	88	2025-01-16	2025-01-21	174.39
17	697	36	2025-01-16	2025-01-17	269.07
18	424	105	2025-01-16	2025-01-18	566.36

Reserva 14 x

controlBD1 GeneradorRegistros* registroprueba mostrartablas mostrarClientes registrosClientes registrosHabitacion registrosreserva registrosServicio x registrosPago revisarCheck



Don't Limit



```
1 • SELECT * FROM Servicio LIMIT 100
```

Result Grid

Filter Rows:

Edits

Export/Import:

Wrap Cell Content:

ServicioID	Nombre	Precio
1	Servicio1	89.75
2	Servicio2	51.58
3	Servicio3	68.66
4	Servicio4	68.58
5	Servicio5	116.91
6	Servicio6	58.82
7	Servicio7	23.38
8	Servicio8	20.43
9	Servicio9	112.01
10	Servicio10	78.78
11	Servicio11	37.84
12	Servicio12	32.85
13	Servicio13	30.74
14	Servicio14	35.17
15	Servicio15	63.60
16	Servicio16	92.51
17	Servicio17	51.76
18	Servicio18	61.28

controlBD1	GeneradorRegistros*	registropueba	mostrartablas	mostrarClientes	registrosClientes	registrosHabitacion	registrosreserva	registrosServicio	registrosPago	revisarCheck
------------	---------------------	---------------	---------------	-----------------	-------------------	---------------------	------------------	-------------------	---------------	--------------

1
2
3

SELECT * FROM Pago LIMIT 100;

PagoID	ReservaID	Monto	MetodoPago	FechaPago
1	855	507.10	Transferencia	2025-01-16 02:03:29
2	965	365.54	Transferencia	2025-01-16 02:03:29
3	768	204.80	Transferencia	2025-01-16 02:03:29
4	231	537.92	Transferencia	2025-01-16 02:03:29
5	972	318.65	Tarjeta	2025-01-16 02:03:29
6	53	320.95	Tarjeta	2025-01-16 02:03:29
7	941	372.29	Transferencia	2025-01-16 02:03:29
8	636	387.93	Transferencia	2025-01-16 02:03:29
9	764	305.06	Efectivo	2025-01-16 02:03:29
10	544	111.75	Efectivo	2025-01-16 02:03:29
11	391	519.15	Tarjeta	2025-01-16 02:03:29
12	589	539.61	Efectivo	2025-01-16 02:03:29
13	705	581.17	Efectivo	2025-01-16 02:03:29
14	937	532.08	Transferencia	2025-01-16 02:03:29
15	268	328.19	Transferencia	2025-01-16 02:03:29
16	730	372.55	Transferencia	2025-01-16 02:03:29
17	629	100.60	Tarjeta	2025-01-16 02:03:29
18	594	403.81	Tarjeta	2025-01-16 02:03:29

En esta imagen podemos observar el check in: Exitoso

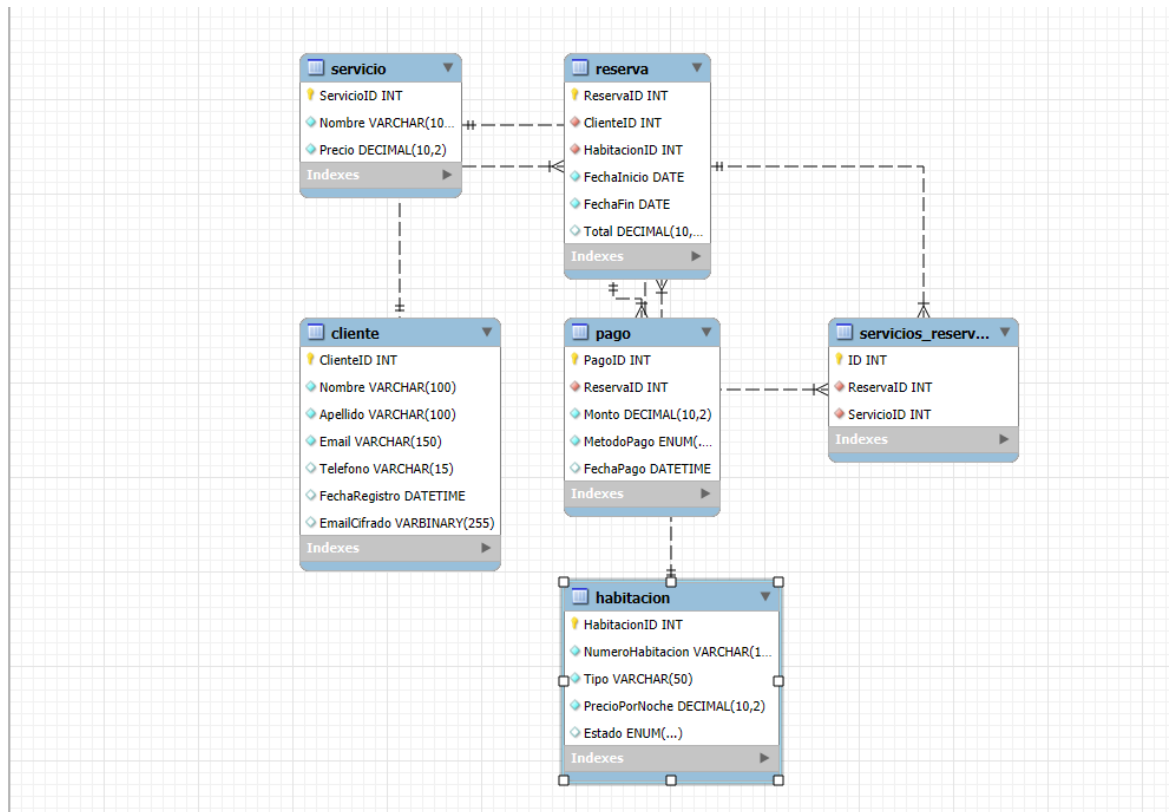
controlBD1	GeneradorRegistros*	registropueba	mostrartablas	mostrarClientes	registrosClientes	registrosHabitacion	registrosreserva	registrosServicio	registrosPago	revisarCheck
------------	---------------------	---------------	---------------	-----------------	-------------------	---------------------	------------------	-------------------	---------------	--------------

1
2
3

CALL CheckInCliente(1, 1, '2025-01-16', '2025-01-18', @Resultado);
SELECT @Resultado;

@Resultado
Check-in exitoso

Diagrama de relación:



1. Tabla Cliente

Propósito: Almacena información sobre los clientes del hotel.

Campos principales:

ClienteID: Identificador único del cliente (clave primaria).

Nombre, Apellido, Email: Información personal del cliente.

Relaciones:

Un cliente puede realizar varias reservas (relación 1:N con la tabla Reserva).

2. Tabla Habitacion

Propósito: Almacena información sobre las habitaciones del hotel.

Campos principales:

HabitacionID: Identificador único de la habitación (clave primaria).

NumeroHabitacion, Tipo, PrecioPorNoche: Detalles de cada habitación.

Estado: Indica si la habitación está disponible, ocupada o en mantenimiento.

Relaciones:

Una habitación puede estar asociada con una reserva (relación 1:N con la tabla Reserva).

3. Tabla Reserva

Propósito: Registra las reservas realizadas por los clientes.

Campos principales:

ReservaID: Identificador único de la reserva (clave primaria).

ClienteID: Relación con el cliente que realizó la reserva.

HabitacionID: Relación con la habitación reservada.

FechaInicio, FechaFin: Periodo de la reserva.

Total: Precio total de la reserva.

Relaciones:

Relación N:1 con la tabla Cliente: Muchas reservas pueden pertenecer a un cliente.

Relación N:1 con la tabla Habitacion: Muchas reservas pueden estar asociadas a una habitación.

Relación 1:N con la tabla Pago: Una reserva puede tener múltiples pagos.

4. Tabla Pago

Propósito: Almacena información sobre los pagos realizados por los clientes.

Campos principales:

PagoID: Identificador único del pago (clave primaria).

ReservaID: Relación con la reserva asociada al pago.

Monto: Cantidad pagada.

MetodoPago: Tipo de pago (tarjeta, efectivo, etc.).

Relaciones:

Relación N:1 con la tabla Reserva: Cada pago está asociado a una única reserva.

5. Tabla Servicio

Propósito: Almacena información sobre los servicios adicionales ofrecidos por el hotel (restaurante, spa, conferencias, etc.).

Campos principales:

ServicioID: Identificador único del servicio (clave primaria).

Nombre, Precio: Descripción y costo del servicio.

Relaciones:

Relación N:M con la tabla Reserva a través de la tabla intermedia

Servicios_Reservas.

6. Tabla Servicios_Reservas (Tabla intermedia)

Propósito: Relaciona las reservas con los servicios adicionales solicitados por los clientes.

Campos principales:

ReservaID: Relación con la reserva.

ServicioID: Relación con el servicio.

Relaciones:

Relación N:1 con la tabla Reserva: Un servicio puede estar asociado a varias reservas.

Relación N:1 con la tabla Servicio: Una reserva puede incluir varios servicios.

Ejemplo Práctico

El cliente Juan Pérez (ClienteID = 1) realiza una reserva (ReservaID = 101) para la habitación 101 (HabitacionID = 1) desde el 15/01/2025 al 17/01/2025.

Durante su estadía, solicita servicios como el spa (ServicioID = 2) y el restaurante (ServicioID = 3).

La reserva genera dos pagos: uno inicial y otro al finalizar la estancia.

Diagrama Resumido

Cliente (1:N) Reserva

Un cliente puede realizar varias reservas.

Habitacion (1:N) Reserva

Una habitación puede estar asociada con varias reservas.

Reserva (1:N) Pago

Una reserva puede generar múltiples pagos.

Servicio (N:M) Reserva

Los servicios adicionales están relacionados con las reservas mediante la tabla intermedia Servicios_Reservas.

Conclusión del proyecto:

Este proyecto de base de datos está diseñado para gestionar de manera eficiente las operaciones de un sistema de reservas de hotel, cubriendo aspectos clave como el manejo de clientes, habitaciones, servicios, reservas y pagos. A continuación, se resumen los puntos más destacados:

- Fortalezas del Diseño
 1. Estructura Relacional Bien Definida:
 2. Cada tabla está diseñada para cumplir con un propósito específico, lo que facilita la organización y el acceso a los datos.
 3. Las relaciones entre tablas mediante claves foráneas aseguran la integridad referencial.
- Escalabilidad:
 1. El uso de campos como AUTO_INCREMENT en claves primarias permite un fácil crecimiento de los datos.
 2. Índices en columnas clave (Email, Estado, FechaInicio) mejoran el rendimiento en consultas.
- Seguridad de Datos:
 1. El cifrado de información sensible, como los emails de los clientes, mediante AES_ENCRYPT protege los datos frente a accesos no autorizados.
 2. Un trigger asegura que los nuevos registros se cifren automáticamente.

- Automatización:
 1. Procedimientos almacenados como CheckInCliente y CheckOutCliente agilizan procesos críticos, reduciendo errores manuales.
 2. Un procedimiento para generar datos de prueba facilita pruebas y simulaciones a gran escala.
- Flexibilidad en Operaciones:
 1. La tabla Servicios_Reservas permite gestionar relaciones muchos a muchos entre servicios y reservas.
 2. La inclusión de opciones como métodos de pago múltiples mejora la funcionalidad del sistema.
- Áreas de Aplicación Práctica
- Gestión Hotelera Integral:

El sistema permite registrar clientes, gestionar reservas, asignar servicios, procesar pagos y actualizar el estado de las habitaciones.
- Análisis de Datos:

Las consultas sobre el uso de habitaciones, ingresos por servicios o análisis de métodos de pago proporcionan información valiosa para la toma de decisiones estratégicas.
- Seguridad y Cumplimiento:

Con el cifrado de datos sensibles, el sistema cumple con estándares de protección de datos, mejorando la confianza del cliente.

Conclusión Final

El proyecto proporciona una solución robusta, eficiente y escalable para la gestión de hoteles, centrada en la seguridad, la funcionalidad y la experiencia del usuario. Este diseño puede ser implementado con éxito en un entorno real y adaptarse a las necesidades específicas de cualquier hotel, desde pequeños negocios familiares hasta cadenas hoteleras internacionales.